

JP11328155

Publication Title:

Symbolic calculation system, symbolic calculation method and parallel circuit simulation system

Abstract:

The coefficient matrix, corresponding to the simultaneous linear equations to be solved, is divided into a plurality of row sets. The row sets as divided are processed in a parallel fashion, and entries specifying the nonzero elements contained in the first to n -th row sets are added to the entry sets E_1 to E_n . Moreover, in regard to each row set, fill-ins which take place at the time of eliminating the i -th variable are obtained in a parallel fashion, and entries specifying the fill-ins are added to the entry sets E_1 to E_n . The coefficient matrix is compressed using those entry sets E_1 to E_n .

Data supplied from the esp@cenet database - <http://ep.espacenet.com>

(19)日本国特許庁(J P)

(12) 公開特許公報 (A)

(11)特許出願公開番号

特開平11-328155

(43)公開日 平成11年(1999)11月30日

(51)Int.Cl.⁸

識別記号

F I

G 0 6 F 17/12

G 0 6 F 15/32A

17/50

15/60

6 6 2 G

審査請求 有 請求項の数5 O L (全 14 頁)

(21)出願番号 特願平10-127738

(22)出願日 平成10年(1998)5月11日

(71)出願人 000004237

日本電気株式会社

東京都港区芝五丁目7番1号

(72)発明者 蜂屋 孝太郎

東京都港区芝五丁目7番1号 日本電気株式会社内

(74)代理人 弁理士 古澤 聡 (外1名)

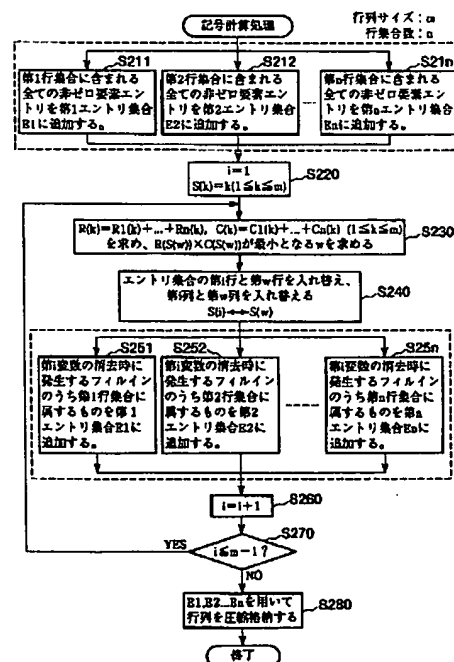
(54)【発明の名称】 記号計算システム及び方法、並びに並列回路シミュレーションシステム

(57)【要約】

【課題】 線形連立方程式の求解における記号計算処理で、扱える係数行列の規模を大きくし、また、計算時間を短縮する。

【解決手段】 求解対象となる線形連立方程式に対応する係数行列を、少なくとも1行ずつからなる複数の行集合に分割する。分割された行集合のそれぞれは、並列に処理されて、第1～第n行集合に含まれている値が

「0」でない要素のエントリが、それぞれに対応するエントリ集合E1～Enに追加される(ステップS211～S21n)。また、第i変数を消去するときに発生するフィルインを行集合毎に並列して求め、そのフィルインに対応するエントリをエントリ集合E1～Enに追加する(ステップS251～S25n)。そして、これらエントリ集合E1～Enのすべてを用いて、係数行列を圧縮する(ステップS280)。



【特許請求の範囲】

【請求項 1】線形連立方程式の求解のための記号計算を行う記号計算システムであって、

求解の対象となる線形連立方程式を示す行列を、それぞれ少なくとも前記行列の 1 行分からなる複数の行集合に分割する行列分割手段と、

前記行列分割手段によって分割された複数の行集合のそれぞれに対応し、対応する行集合に含まれる値が 0 でない要素のエントリを、それぞれに対応するエントリ集合に追加する複数の第 1 のエントリ追加手段と、

値が 1 から前記行列の行数まで順次変化される第 1 の変数が示す変数の消去時に必要となる演算回数が最小となる第 2 の変数を順次求めるピボット選択手段と、

前記行列中の前記第 1 の変数が示す行と前記第 2 の変数が示す行、及び前記行列中の前記第 1 の変数が示す列と前記第 2 の変数が示す列とをそれぞれ交換するピボット交換手段と、

前記複数の第 1 のエントリ追加手段のそれぞれに対応し、前記ピボット交換手段によって行及び列が交換された行列について変数の消去時に発生するフィルインを対応する行集合毎に求め、該フィルインのエントリを対応する第 1 のエントリ追加手段でそれぞれエントリが追加されたエントリ集合にさらに追加する複数の第 2 のエントリ追加手段と、

前記第 1、第 2 のエントリ追加手段によってエントリが追加された複数のエントリ集合を用いて、前記行列を圧縮する行列圧縮手段と、を備えることを特徴とする記号計算システム。

【請求項 2】前記複数の第 1、第 2 のエントリ追加手段は、対応するもの同士で同一の手段によって構成され、前記複数の第 1、第 2 のエントリ追加手段はそれぞれ、対応する行集合の行内エントリ数を記憶する行内エントリ数配列を管理する手段と、列内エントリ数を記憶する列内エントリ数配列を管理する手段とを有し、前記ピボット選択手段は、前記行内エントリ数配列が記憶する行内エントリ数をそれぞれ配列の添字が等しいもの同士で加算する手段と、前記列内エントリ数配列が記憶する列内エントリ数をそれぞれ配列の添字が等しいもの同士で加算する手段と、これらの手段によってそれぞれ加算された行内エントリ数と列内エントリ数とを添字が等しいもの同士で乗算する手段と、これらの乗算結果中の値が最も小さいものの添字を選択する選択手段とを有し、前記選択手段によって選択された添字を前記第 2 の変数とすることを特徴とする請求項 1 に記載の記号計算システム。

【請求項 3】前記ピボット交換手段は、行列中の交換された行及び列を示す置換配列を記憶する手段と、該手段に記憶された置換配列を前記複数の第 2 のエントリ追加手段のそれぞれに供給する手段とを備え、前記複数の第 2 のエントリ追加手段は、前記ピボット交

換手段から供給された置換配列に従って、行及び列が交換された行列について変数の消去時に発生するフィルインを対応する行集合毎に求めることを特徴とする請求項 1 または 2 に記載の記号計算システム。

【請求項 4】線形連立方程式の求解のための記号計算を行う記号計算方法であって、

求解の対象となる線形連立方程式を示す行列を、それぞれ少なくとも前記行列の 1 行分からなる複数の行集合に分割する行列分割ステップと、

10 前記行列分割ステップで分割された複数の行集合のそれぞれに応じて並列して動作し、対応する行集合に含まれる値が 0 でない要素のエントリを、それぞれに対応するエントリ集合に追加する第 1 のエントリ追加ステップと、

値が 1 から前記行列の行数まで順次変化される第 1 の変数が示す変数の消去時に必要となる演算回数が最小となる第 2 の変数を順次求めるピボット選択ステップと、

前記行列中の前記第 1 の変数が示す行と前記第 2 の変数が示す行、及び前記行列中の前記第 1 の変数が示す列と前記第 2 の変数が示す列とをそれぞれ交換するピボット交換ステップと、

20 前記第 1 のエントリ追加ステップで並列に行われるそれぞれの処理に対応して並列に動作し、前記ピボット交換ステップで行及び列が交換された行列について変数の消去時に発生するフィルインを対応する行集合毎に求め、該フィルインのエントリを第 1 のエントリ追加ステップでそれぞれエントリが追加されたエントリ集合にさらに追加する第 2 のエントリ追加ステップと、

30 前記第 1、第 2 のエントリ追加ステップでエントリが追加された複数のエントリ集合を用いて、前記係数行列を圧縮する行列圧縮ステップと、を備えることを特徴とする記号計算方法。

【請求項 5】シミュレーションの対象となる回路を複数の部分回路に分割する部分回路分割手段と、

前記部分回路分割手段で分割された複数の部分回路のそれぞれに対応し、対応する部分回路の回路方程式に対応する部分回路行列を求める複数の部分回路行列計算手段と、

40 前記複数の部分回路行列計算手段によって計算された部分回路行列のそれぞれに記号計算を行う複数の部分回路記号計算手段と、

前記複数の記号計算手段によって記号計算されたそれぞれの結果にさらに数値計算処理を行い、対応する部分回路の回路方程式の解を求める複数の部分回路数値計算手段と、

前記複数の部分回路数値計算手段によって求められた各部分回路の回路方程式の解に従って、部分回路を結合した結合回路の回路方程式に対応する結合回路行列を求める結合回路行列計算手段と、

50 前記結合回路行列計算手段によって計算された結合回路

行列に記号計算を行う結合回路記号計算手段と、
前記結合回路記号計算手段によって記号計算された結果
に更に数値計算処置を行い、結合回路の回路方程式の解
を求める結合回路数値計算手段と、を備え、

前記結合回路数値計算手段は、

前記結合回路行列計算手段によって求められた結合回路
行列を、それぞれ少なくとも前記行列の1行分からなる
複数の行集合に分割する行列分割手段と、

前記行列分割手段によって分割された複数の行集合のそ
れぞれに対応し、対応する行集合に含まれる値が0でない
要素のエントリを、それぞれに対応するエントリ集合
に追加する複数の第1のエントリ追加手段と、

値が1から前記結合回路行列の行数まで順次変化される
第1の変数が示す変数の消去時に必要となる演算回数が
最小となる第2の変数を順次求めるピボット選択手段
と、

前記結合回路行列中の前記第1の変数が示す行と前記第
2の変数が示す行、及び前記結合回路行列中の前記第1
の変数が示す列と前記第2の変数が示す列とをそれぞれ
交換するピボット交換手段と、

前記複数の第1のエントリ追加手段のそれぞれに対応
し、前記ピボット交換手段によって行及び列が交換され
た結合回路行列について変数の消去時に発生するフ
ィルインを対応する行集合毎に求め、該フィルインのエ
ントリを対応する第1のエントリ追加手段でそれぞれエ
ントリが追加されたエントリ集合にさらに追加する複数
の第2のエントリ追加手段と、

前記第1、第2のエントリ追加手段によってエントリが
追加された複数のエントリ集合を用いて、前記結合回路
行列を圧縮する行列圧縮手段と、を備えることを特徴と
する並列回路シミュレーションシステム。

【発明の詳細な説明】

【0001】

【発明の属する技術分野】本発明は、線形連立方程式の
解を求めるための記号計算を行う記号計算システム及び
方法、並びに線形連立方程式の解を求めることにより回
路のシミュレーションを行う並列回路シミュレーション
システムに関する。

【0002】

【従来の技術】線形連立方程式の解を求めるための処理
は、一般に、記号計算処理と数値計算処理とからなる。
この線形連立方程式の求解処理における記号計算処理の
一例が、“Fundamentals of Computer-Aided Circuit Si
mulation” (William J. McCalla著、1988年刊)の
第3章“Sparse Matrix Methods”に記載されている。

【0003】図9は、上記従来例の線形連立方程式の求
解処理における記号計算処理を示すフローチャートであ
る。まず、求解の対象となる線形連立方程式を表す係数
行列全体に対して1つのエントリ集合Eを用意し、係数
行列中の値が「0」でない要素のエントリをエントリ集

合Eに順次追加していく。ここで、第k行に含まれるエ
ントリ数及び第k列に含まれるエントリ数を示す配列R
(k)、C(k)も係数行列全体に対して1つずつ用意
して、エントリ登録時に値を更新していく(ステップS
410)。

【0004】次に、何番目の変数を消去するかを示す変
数iに「1」を代入し、初期化する(ステップS42
0)。次に、 $R(w) \times C(w)$ が最小となるwの値を
求める。ここで、 $R(w)$ 、 $C(w)$ の値を求めるとき
に、既に消去された変数と式に対応する行列部分のエ
ントリや、対角要素となるエントリ数はカウントされない
(ステップS430)。

【0005】wの値が求められると、エントリ集合Eの
第i行と第w行とを交換し、また、第i列と第w列とを
交換する(ステップS440)。次に、第i変数の消去
時に発生するフィルインを求め、エントリ集合Eに追加
する(ステップS450)。

【0006】ステップS450の処理が終了すると、変
数iの値を「1」だけインクリメントされ(ステップS
460)、さらに変数iの値が $m-1$ (m:係数行列の
サイズ)以下であるかどうか判定される(ステップS
470)。ステップS470で変数iの値が $m-1$ 以下
であると判定されているときは、ステップS430の処
理に戻る。このような処理の繰り返しによって、最適な
ピボット順序とフィルインの発生位置が求められる。

【0007】一方、ステップS470で変数iの値がm
の値に等しくなると判定されると、ステップS410
及びステップS450でエントリが追加されたエントリ
集合Eを用いて行列を圧縮し、圧縮した行列データを格
納する(ステップS480)。そして、このフローチャ
ートの処理を終了する。そして、数値計算処理に進むこ
ととなる。

【0008】

【発明が解決しようとする課題】しかしながら、上記従
来例には、次のような問題点があった。まず第1に、記
号計算処理を分散メモリ型の並列コンピュータで実行す
る場合に、記号計算処理を単一のノード上で逐次実行し
なければならなかった。このため、ノード内の局所メモ
リの容量によって、扱える係数行列のサイズが制限され
てしまうという問題点があった。

【0009】また、第2に、係数行列のサイズ(行及び
列のそれぞれの数(行数及び列数は一致する))をmと
した場合の記号計算処理の時間は、mの1乗から3乗程
度のオーダで増加するため、係数行列のサイズが大き
くなり、エントリ集合に追加するエントリ数が膨大にな
ると、記号計算処理を実用的な時間内で終了することが
できなくなってしまうという問題点があった。

【0010】このような問題点は、線形連立方程式の解
を求めることによって回路のシミュレーションを行う場
合にも生じていた。

【0011】本発明は、上記従来例の問題点を解消するためになされたものであり、扱える行列の規模が大きく、計算時間が短い記号計算システム及び方法を提供することを目的とする。

【0012】本発明は、また、大規模な回路のシミュレーションを高速に行うことができる並列回路シミュレーションシステムを提供することを目的とする。

【0013】

【課題を解決するための手段】上記目的を達成するため、本発明の第1の実施の形態にかかる記号計算システムは、線形連立方程式の求解のための記号計算を行う記号計算システムであって、求解の対象となる線形連立方程式を示す行列を、それぞれ少なくとも前記行列の1行分からなる複数の行集合に分割する行列分割手段と、前記行列分割手段によって分割された複数の行集合のそれぞれに対応し、対応する行集合に含まれる値が0でない要素のエントリを、それぞれに対応するエントリ集合に追加する複数の第1のエントリ追加手段と、値が1から前記行列の行数まで順次変化される第1の変数が示す変数の消去時に必要となる演算回数が最小となる第2の変数を順次求めるピボット選択手段と、前記行列中の前記第1の変数が示す行と前記第2の変数が示す行、及び前記行列中の前記第1の変数が示す列と前記第2の変数が示す列とをそれぞれ交換するピボット交換手段と、前記複数の第1のエントリ追加手段のそれぞれに対応し、前記ピボット交換手段によって行及び列が交換された行列について変数の消去時に発生するフィルインを対応する行集合毎に求め、該フィルインのエントリを対応する第1のエントリ追加手段でそれぞれエントリが追加されたエントリ集合にさらに追加する複数の第2のエントリ追加手段と、前記第1、第2のエントリ追加手段によってエントリが追加された複数のエントリ集合を用いて、前記行列を圧縮する行列圧縮手段と、を備えることを特徴とする。

【0014】上記計算システムによれば、エントリ集合へのエントリの追加を複数の第1、第2のエントリ追加手段によって並列して行うことができる。このため、線形連立方程式の解を求める場合の記号計算を高速に行うことができる。

【0015】また、上記計算システムを分散メモリ型の並列コンピュータで実現する場合には、複数の第1、第2のエントリ追加手段を実現するためのノードが有する局所メモリに記憶すべきデータ量が、1つのノードでエントリ集合へのエントリ追加を行った場合に比べて、ほぼ行列の分割数の逆数をかけたものとすることができる。このため、各ノードが有する局所メモリの容量によって、扱える行列のサイズが制限されることが少なくなり、大規模な線形連立方程式の求解処理も行えるようになる。

【0016】上記記号計算システムにおいて、前記複数の

の第1、第2のエントリ追加手段は、対応するもの同士で同一の手段、例えば、並列コンピュータの同一のノードによって構成されたものとして行うことができる。この場合、前記複数の第1、第2のエントリ追加手段はそれぞれ、対応する行集合の行内エントリの数を記憶する行内エントリ数配列を管理する手段と、列内エントリの数を記憶する列内エントリ数配列を管理する手段とを有するものででき、前記ピボット選択手段は、前記行内エントリ数配列が記憶する行内エントリ数をそれぞれ配列の添字が等しいもの同士で加算する手段と、前記列内エントリ数配列が記憶する列内エントリ数をそれぞれ配列の添字が等しいもの同士で加算する手段と、これらの手段によってそれぞれ加算された行内エントリ数と列内エントリ数とを添字が等しいもの同士で乗算する手段と、これらの乗算結果中の値が最も小さいものの添字を選択する選択手段とを有し、前記選択手段によって選択された添字を前記第2の変数とすることができる。

【0017】また、上記記号計算システムにおいて、前記ピボット交換手段は、行列中の交換された行及び列を示す置換配列を記憶する手段と、該手段に記憶された置換配列を前記複数の第2のエントリ追加手段のそれぞれに供給する手段とを備えるものとしてもよい。前記複数の第2のエントリ追加手段は、前記ピボット交換手段から供給された置換配列に従って、行及び列が交換された行列について変数の消去時に発生するフィルインを対応する行集合毎に求めるものとして行うことができる。

【0018】このような置換配列を用いることによって、行列或いは行集合のやり取りをしなくても、第2のエントリ追加手段がフィルインを求めることができる。

【0019】上記目的を達成するため、本発明の第2の観点にかかる記号計算方法は、線形連立方程式の求解のための記号計算を行う記号計算方法であって、求解の対象となる線形連立方程式を示す行列を、それぞれ少なくとも前記行列の1行分からなる複数の行集合に分割する行列分割ステップと、前記行列分割ステップで分割された複数の行集合のそれぞれに応じて並列して動作し、対応する行集合に含まれる値が0でない要素のエントリを、それぞれに対応するエントリ集合に追加する第1のエントリ追加ステップと、値が1から前記行列の行数まで順次変化される第1の変数が示す変数の消去時に必要となる演算回数が最小となる第2の変数を順次求めるピボット選択ステップと、前記行列中の前記第1の変数が示す行と前記第2の変数が示す行、及び前記行列中の前記第1の変数が示す列と前記第2の変数が示す列とをそれぞれ交換するピボット交換ステップと、前記第1のエントリ追加ステップで並列に行われるそれぞれの処理に対応して並列に動作し、前記ピボット交換ステップで行及び列が交換された行列について変数の消去時に発生するフィルインを対応する行集合毎に求め、該フィルインのエントリを第1のエントリ追加ステップでそれぞれエ

ントリが追加されたエントリ集合にさらに追加する第2のエントリ追加ステップと、前記第1、第2のエントリ追加ステップでエントリが追加された複数のエントリ集合を用いて、前記係数行列を圧縮する行列圧縮ステップと、を備えることを特徴とする。

【0020】上記目的を達成するため、本発明の第3の観点にかかる並列回路シミュレーションシステムは、シミュレーションの対象となる回路を複数の部分回路に分割する部分回路分割手段と、前記部分回路分割手段で分割された複数の部分回路のそれぞれに対応し、対応する部分回路の回路方程式に対応する部分回路行列を求める複数の部分回路行列計算手段と、前記複数の部分回路行列計算手段によって計算された部分回路行列のそれぞれに記号計算を行う複数の部分回路記号計算手段と、前記複数の記号計算手段によって記号計算されたそれぞれの結果にさらに数値計算処理を行い、対応する部分回路の回路方程式の解を求める複数の部分回路数値計算手段と、前記複数の部分回路数値計算手段によって求められた各部分回路の回路方程式の解に従って、部分回路を結合した結合回路の回路方程式に対応する結合回路行列を求める結合回路行列計算手段と、前記結合回路行列計算手段によって計算された結合回路行列に記号計算を行う結合回路記号計算手段と、前記結合回路記号計算手段によって記号計算された結果に更に数値計算処理を行い、結合回路の回路方程式の解を求める結合回路数値計算手段と、を備え、前記結合回路数値計算手段は、前記結合回路行列計算手段によって求められた結合回路行列を、それぞれ少なくとも前記行列の1行分からなる複数の行集合に分割する行列分割手段と、前記行列分割手段によって分割された複数の行集合のそれぞれに対応し、対応する行集合に含まれる値が0でない要素のエントリを、それぞれに対応するエントリ集合に追加する複数の第1のエントリ追加手段と、値が1から前記結合回路行列の行数まで順次変化される第1の変数が示す変数の消去時に必要となる演算回数が最小となる第2の変数を順次求めるピボット選択手段と、前記結合回路行列中の前記第1の変数が示す行と前記第2の変数が示す行、及び前記結合回路行列中の前記第1の変数が示す列と前記第2の変数が示す列とをそれぞれ交換するピボット交換手段と、前記複数の第1のエントリ追加手段のそれぞれに対応し、前記ピボット交換手段によって行及び列が交換された結合回路行列について変数の消去時に発生するフィルインに対応する行集合毎に求め、該フィルインのエントリに対応する第1のエントリ追加手段でそれぞれエントリが追加されたエントリ集合にさらに追加する複数の第2のエントリ追加手段と、前記第1、第2のエントリ追加手段によってエントリが追加された複数のエントリ集合を用いて、前記結合回路行列を圧縮する行列圧縮手段と、を備えることを特徴とする。

【0021】

【発明の実施の形態】以下、添付図面を参照して、本発明の実施の形態について説明する。

【0022】「第1の実施の形態」図1は、この実施の形態に適用される並列コンピュータの構成を示すブロック図である。

【0023】図示するように、この並列コンピュータ1は、相互結合網10を介して結合された複数のノード11～11から構成されている。ノード11～11は、それぞれ要素プロセッサ11a～11aと、局所メモリ11b～11bを備える。要素プロセッサ11a～11aと局所メモリ11b～11bとは、それぞれバス11c～11cを介して接続されており、また、バス11c～11cは、相互結合網10に接続されている。

【0024】要素プロセッサ11a～11aは、相互結合網10を介して互いにメッセージを交換し合う。また、要素プロセッサ11a～11aは、相互結合網10を介して他のノードが有する局所メモリ11b～11bに遠隔アクセスする。これらによって、並列コンピュータ1は、あるノードから他の1つ以上のノードへ処理を依頼し、依頼した処理の並列処理を可能としている。

【0025】局所メモリ11b～11bはそれぞれ、対応する要素プロセッサ11a～11aが実行するプログラムを記憶すると共に、要素プロセッサ11a～11aのワークエリアとして使用される。

【0026】並列コンピュータ1の相互結合網10には、入力装置21及び外部記憶装置22を有するワークステーション2が接続されている。ワークステーション2の入力装置21は、計算対象となる線形連立方程式（或いはその係数行列）を入力する。また、ワークステーション1は、係数行列を並列コンピュータ1に渡し、線形連立方程式の求解処理を依頼する。

【0027】以下、この実施の形態において、図1の並列コンピュータで実行される線形連立方程式の求解処理について、説明する。図2は、この実施の形態にかかる線形連立方程式の求解処理の概略を示すフローチャートである。

【0028】線形連立方程式の求解処理は、エントリ登録処理（ステップS110）と、対角ピボット処理（ステップS120）と、行列の圧縮格納処理（ステップS130）と、LU分割処理（ステップS140）と、前進代入・後進代入処理（ステップS150）とを順次実行することによって行われる。これらのうちで、ステップS110～S130までが記号計算処理であり、ステップS140とS150とが数値計算処理である。

【0029】ステップS110のエントリ登録処理では、ワークステーション2から渡された係数行列のうちの値が「0」でない要素の位置がエントリ集合Eに追加される。この処理は、後述する行集合毎に並列して行われるものであり、初めに行集合毎のエントリ集合Eを空

集合とし、係数行列中の値が「0」でない要素の行 i と列 j とのエントリを i, j をそれぞれのエントリ集合 E に追加するという処理をすべての要素に対して行うものである。

【0030】ステップS120の対角ピボット処理では、ステップS140のLU分解処理での演算回数ができるだけ少なくなるように、係数行列の行交換と列交換とが行われる。この処理には、元の係数行列では値が「0」であった要素でLU分解処理の過程で値が

「0」でなくなる要素（ファルイン）のエントリを行集合毎にエントリ集合 E に追加する処理が含まれている。

【0031】ステップS130の行列の圧縮格納処理では、ステップS120までの処理で得られたエントリ集合 E を用いて、係数行列及びLU分解後に得られる行列の値が「0」でない要素の値のみを格納するための領域が確保され、係数行列の要素の値が格納される。

【0032】ステップS140のLU分割処理では、係数行列を左下三角行列 L と右上三角行列 U との積に因数分解する処理が行われる。ここでは、係数行列が格納された領域に上書きをしていくことによって処理が進められる。

【0033】ステップS150の前進代入・後進代入処理では、係数行列で表される線形連立方程式の解が求められる。この処理では、例えば、係数行列で表される線形連立方程式が $Ax=b$ （ A ：係数行列、 x ：解ベクトル、 b ：右辺ベクトル）としたときに、 $Ly=b$ を解いて y の解を求める前進代入処理と、 $Ux=y$ を解いて x の解を求める後進代入処理とが行われる。

【0034】図3は、図2の記号計算処理（ステップS110～S130）を詳細に示すフローチャートである。このフローチャートにおけるステップS211～S211が図2のステップS110に対応し、ステップS220～S270が図2のステップS120に対応し、ステップS280が図2のステップS130に対応する。

【0035】ユーザは、ワークステーション2の入力装置21を操作して、解を求める対象となる線形連立方程式を入力する。そして、入力装置21からワークステーション2に入力した線形連立方程式の解を求めるための指示を入力する。すると、ワークステーション2は、線形連立方程式の係数行列（行列サイズを m とする）を求め、さらに複数の行集合（集合数を n （ $n \leq 1$ ）とする）に分ける。そして、ワークステーション2は、この係数行列を並列コンピュータ1に渡すと共に、記号計算処理の依頼をする。

【0036】並列コンピュータ1のノード11～11の要素プロセッサ11a～11aうち、記号計算処理の依頼を受け取ったノード（ノード11とする）の要素プロセッサ11aは、渡された係数行列を行集合毎にノード11～11中の n 個のノードに相互結合網10を介して

送信する（但し、1つは自ノード11内でプロセス間通信によって送信してもよい）。

【0037】係数行列中の行集合を受け取った n 個のノード（ノード11～11とする）は、それぞれが並列に動作する。これらのノード11～11の要素プロセッサ11a～11aは、それぞれ行集合毎に要素を空集合とするエントリ集合 $E1 \sim En$ を用意し、自ノード内の局所メモリ11b～11bに領域を確保する。また、第 k 行に含まれるエントリ数及び第 k 列に含まれるエントリ数を示す配列 $R1(k) \sim Rn(k)$ 、 $C1(k) \sim Cn(k)$ を行集合毎に用意し、局所メモリ11b～11bに領域を確保する。

【0038】そして、要素プロセッサ11a～11aは、それぞれ第1～第 n 行集合に含まれている値が「0」でない要素のエントリを、それぞれの局所メモリ11b～11bに領域を確保したエントリ集合 $E1 \sim En$ に追加していく。また、要素プロセッサ11a～11aは、エントリの追加時には、配列 $R1(k) \sim Rn(k)$ 、 $C1(k) \sim Cn(k)$ の値を更新していく（ステップS211～S21n）。

【0039】 n 個のノード11～11においてステップS211～S21nの処理が終了すると、並列コンピュータ1の1個のノード11～11のうちの1つのノード（ノード11とする）の要素プロセッサ11aは、何番目の変数を消去するかを示す変数 i を用意し、「1」を代入して初期化する。さらに、要素プロセッサ11aは、後述するステップS240の処理で交換された行及び列の数を示す置換配列 $S(k)$ を用意し、局所メモリ11bに領域を確保する。この時点において、置換配列 $S(k)$ は行及び列が交換されていない状態で初期化される（ステップS220）。

【0040】次に、要素プロセッサ11aは、ノード11～11の局所メモリ11b～11bに遠隔アクセスし、配列 $R1(k) \sim Rn(k)$ 、 $C1(k) \sim Cn(k)$ を取得し、それぞれの要素を加算した配列 $R(k)$ 、 $C(k)$ を求める。また、要素プロセッサ11aは、ノード11～11の局所メモリ11b～11bに遠隔アクセスし、エントリ集合 $E1 \sim En$ を取得し、その和集合であるエントリ集合 E を求める。

【0041】次に、要素プロセッサ11aは、変数を消去するときに必要となる演算回数が最小となる変数を、まだ消去していない変数から求める。すなわち、 k 番目の変数を消去するときに必要となる演算回数は、 $R(k) \times C(k)$ によって求められるので、要素プロセッサ11aは、 $R(S(w)) \times C(S(w))$ が最小となる w の値を求める。ここで、 $R(S(w))$ 、 $C(S(w))$ の値を求めるときに、既に消去された変数と式に対応する行列部分のエントリや、対角要素となるエントリ数はカウントされない（ステップS230）。

【0042】要素プロセッサ11aは、エントリ集合 E

の第 i 行と第 w 行とを交換し、第 i 列と第 w 列とを交換する。この交換は、 $S(i)$ の値と $S(w)$ の値とを交換することによって行われる(ステップS240)。

【0043】要素プロセッサ11aは、次に、ステップS240の結果求められた置換配列 $S(k)$ をメッセージとして相互結合網10を介してノード11~1nに送信する。置換配列 $S(k)$ を受け取ったノード11~1nは、再びそれぞれが並列に動作する。これらのノードの要素プロセッサ11a~1naはそれぞれ、第 i 変数を消去するときに発生するフィルインを行集合毎に求め、そのフィルインに対応するエントリをエントリ集合 $E1 \sim En$ に追加する。

【0044】ここで、右上三角行列の第 i 行に含まれるすべてのエントリの位置がわかれば、各行集合で発生するフィルインの位置は、行集合毎に並列に求めることができる。従って、フィルインに対応するエントリをエントリ集合 $E1 \sim En$ のそれぞれに追加する処理も並列に行うことが可能となる(ステップS251~S25n)。

【0045】ノード11~1nにおいて、ステップS251~S25nの処理が終了すると、ノード11の要素プロセッサ11aは、変数 i の値を「1」だけインクリメントする(ステップS260)。要素プロセッサ11aは、さらに変数 i の値が $m-1$ の値以下であるかどうかを判定する(ステップS270)。

【0046】ステップS270で変数 i の値が $m-1$ 以下であると判定されているときは、ステップS230の処理に戻る。このような処理の繰り返しによって、最適なピボット順序とフィルインの発生位置が求められる。

【0047】一方、ステップS270で変数 i の値が m の値に等しくなると判定されると、要素プロセッサ11aは、ノード11~1nの局所メモリ11b~1nbに遠隔アクセスし、エントリ集合 $E1 \sim En$ を取得する。要素プロセッサ11aは、エントリ集合 $E1 \sim En$ を用いて行列を圧縮し、圧縮した行列データを局所メモリ11bに格納する(ステップS280)。

【0048】ここで、行列データの圧縮形式としては、例えば、“Templates for the Solution of Linear Systems: Building Blocks for Iterative Methods”(R. Barrett他著、1994年刊)に記載されているような行圧縮形式を用いることができる。

【0049】ステップS280の処理が終了すると、記号計算処理が終了し、次に、数値計算処理が実行される。そして、ワークステーション2から並列コンピュータ1に係数行列として渡された線形連立方程式の解が求められる。

【0050】以下、この実施の形態における線形連立方程式の求解処理の具体例について、図4~図7を参照して説明する。ここで、ワークステーション2から並列コンピュータ1に渡される係数行列は、図4に示すもので

あり、図中の破線で示すように3つの行集合に分けられているものとする。

【0051】各行集合を受け取ったノード(ノード11~13とする)では、ステップS211~S213において行集合毎に値が「0」でないエントリがエントリ集合 $E1 \sim E3$ に追加される。このとき局所メモリ11b~13bに記憶されるエントリ集合 $E1 \sim E3$ は、図5(a)に示すように、

$E1 = \{ \langle 1, 1 \rangle, \langle 1, 3 \rangle, \langle 1, 5 \rangle, \langle 2, 2 \rangle, \langle 2, 3 \rangle, \langle 2, 4 \rangle, \langle 2, 6 \rangle \}$
 $E2 = \{ \langle 3, 1 \rangle, \langle 3, 2 \rangle, \langle 3, 3 \rangle, \langle 4, 2 \rangle, \langle 4, 4 \rangle, \langle 4, 5 \rangle, \langle 4, 6 \rangle \}$
 $E3 = \{ \langle 5, 1 \rangle, \langle 5, 4 \rangle, \langle 5, 5 \rangle, \langle 6, 2 \rangle, \langle 6, 4 \rangle, \langle 6, 6 \rangle \}$

に設定される。

【0052】また、局所メモリ11b~13bに記憶されている、エントリ集合 $E1 \sim E3$ へのエントリの追加時に更新されていく配列 $R1 \sim R3$ 、 $C1 \sim C3$ は、図5(a)に示すように、

$R1 = [2, 3, 0, 0, 0, 0]$
 $C1 = [0, 0, 2, 1, 1, 1]$
 $R2 = [0, 0, 2, 3, 0, 0]$
 $C2 = [1, 2, 0, 0, 1, 1]$
 $R3 = [0, 0, 0, 0, 2, 2]$
 $C3 = [1, 1, 0, 2, 0, 0]$

となる。

【0053】そして、ステップS220において、ノード11の局所メモリ11bに記憶される変数 i 及び置換配列 S が初期化され、図5(a)に示すように、

$i = 1$
 $S = [1, 2, 3, 4, 5, 6]$

に設定される。

【0054】次に、ステップS230において、ピボット選択処理が行われる。ここで、

$R = [2, 3, 2, 3, 2, 2]$
 $C = [2, 3, 2, 3, 2, 2]$

であるので、 $R(S(w)) \times C(S(w))$ を最小にする w の値は、1となる。

【0055】次に、ステップS240において、 $S(i)$ と $S(w)$ とを入れ替える処理が行われるが、 $i = w$ であるので、実際には変化はない。もともと、置換配列 S をノード11~13のすべてにメッセージとして送信する処理は行われる。

【0056】次に、ステップS251~S253において、元の係数行列における $S(i)$ 番目の変数を消去するときに生じるフィルインに対応するエントリが、ノード11~13のそれぞれでエントリ集合 $E1 \sim E3$ に追加される。このとき、局所メモリ11b~13bに記憶されるエントリ集合 $E1 \sim E3$ は、図5(b)に示すように、

E1 = {<1, 1>, <1, 3>, <1, 5>, <2, 2>, <2, 3>, <2, 4>, <2, 6>},
 E2 = {<3, 1>, <3, 2>, <3, 3>, <3, 5>, <4, 2>, <4, 4>, <4, 5>, <4, 6>},
 E3 = {<5, 1>, <5, 3>, <5, 4>, <5, 5>, <6, 2>, <6, 4>, <6, 6>}

に設定される。

【0057】また、配列R1~R3、C1~C3の値も更新され、図5(b)に示すように、

R1 = [0, 3, 0, 0, 0, 0]

C1 = [0, 0, 1, 1, 0, 1]

R2 = [0, 0, 2, 3, 0, 0]

C2 = [1, 2, 0, 0, 1, 1]

R3 = [0, 0, 0, 0, 2, 2]

C3 = [0, 1, 1, 2, 0, 0]

となる。

【0058】そして、ステップS260において変数iがインクリメントされ、図5(b)に示すように、i = 2となる。そして、再びステップS230の処理が実行される。

【0059】以上のような処理をi = 6 - 1 = 5となるまで繰り返す。この過程において、ステップS251~S253では、図6に示すようにフィルインに対応するエントリがエントリ集合E1~E3に追加されていく。i = 5となったとき、エントリ集合E1~E3、配列R1~R3、C1~C3、変数i、及び置換配列Sは、図5(c)に示すように、次の通りとなる。

【0060】E1 = {<1, 1>, <1, 3>, <1, 5>, <2, 2>, <2, 3>, <2, 4>, <2, 5>, <2, 6>}

E2 = {<3, 1>, <3, 2>, <3, 3>, <3, 5>, <4, 2>, <4, 4>, <4, 5>, <4, 6>}

E3 = {<5, 1>, <5, 2>, <5, 3>, <5, 4>, <5, 5>, <6, 2>, <6, 4>, <6, 6>}

【0061】R1 = [0, 0, 0, 0, 0, 0]

C1 = [0, 0, 0, 0, 0, 0]

R2 = [0, 0, 0, 0, 0, 0]

C2 = [0, 0, 0, 0, 0, 0]

R3 = [0, 0, 0, 0, 0, 0]

C3 = [0, 0, 0, 0, 0, 0]

S = [1, 3, 5, 4, 2, 6]

【0062】そして、上記の処理で最終的に得られたエントリ集合E1~E3を用いて、ステップS280において、図7に示すように行列データが圧縮されて、ノード11の局所メモリ11bに格納される。

【0063】上記の処理でエントリ集合Eに追加されるエントリ数は、図5(c)に示すように、最終的には2

4個となっているが、行集合毎、すなわちノード毎に8個ずつに分散されている。従って、各ノード11~13におけるエントリ集合E1~E3のための局所メモリ11b~13bの使用量が、1つのノードで処理を行う場合に比べて3分の1で済むことがわかる。また、各ノード11~13は、係数行列全体を保持する必要がなく、その3分の1のデータ量で済む行集合を保持していればよい。従って、1つのノードにおける局所メモリの容量が従来のほぼ3分の1で済むことになり、また、局所メモリの容量が変わらないのであれば、従来よりもサイズの大きな係数行列（要素数がほぼ3倍）を扱えることとなる。

【0064】また、図6に示すように、ステップS251~S253のエントリの追加登録の処理は、ノード11~13で並列処理を行っていることにより、全体として9ステップで済んでいる。これに対して、1つのノードで同様の処理を行った場合には、追加登録の総数が17個であるので、17ステップを要することとなる。すなわち、この部分での計算速度がほぼ2倍になっていることがわかる。

【0065】以上説明したように、この実施の形態によれば、線形連立方程式の求解処理の最初の段階として実行される記号計算処理において、並列コンピュータ1の複数のノード11~1nが並列処理を行っている。

【0066】このとき、各ノード11~1nは、局所メモリ11a~1na内に係数行列を分割した行集合のみを保持すればよく、局所メモリ11a~1naに記憶されるまた、エントリ集合E1~E3も行集合に対応するもののみで足りることとなる。このため、同一の係数行列に対して計算を行うならば、局所メモリ11b~1ncの容量が従来例よりも少なく済む。また、従来例と局所メモリ11b~1ncの容量が同一であるならば、従来例よりも大きなサイズの係数行列を扱うことが可能となる。

【0067】また、エントリ集合E1~E3へのエントリへの追加を、複数のノードで分散して行うことができる。このため、従来例に比べてエントリの追加に要する計算時間が短くて済むようになる。

【0068】また、ノード11から各ノード11~1nへ置換配列Sを送ることによって、ノード間で行列或いは行集合のやり取りをしなくても、各ノード11~1nでフィルインを求めることが可能となる。

【0069】[第2の実施の形態] この実施の形態では、線形連立方程式の解を求めることによる並列回路のシミュレーションに、本発明を適用した場合について説明する。

【0070】この実施の形態においても、並列回路シミュレーションを行うハードウェアとして、図1に示す並列コンピュータ1を用いることができる。但し、この実施の形態において、シミュレーション対象となる並列回

10

20

30

40

50

路の回路接続情報は、ワークステーション2の外部記憶装置22にライブラリとして登録されており、入力装置21からの指示に従って、ライブラリから読み出されて並列コンピュータ1に投入される。

【0071】以下、この実施の形態において、図1の並列コンピュータ1で実行される並列回路シミュレーションの処理を、図9のフローチャートを参照して説明する。

【0072】処理が開始すると、まず、ユーザは入力装置21を操作して、外部記憶装置22にライブラリとして登録されている回路接続情報をワークステーション2から相互結合網10を介して並列コンピュータ1に投入する（ステップS300）。

【0073】並列コンピュータ1のノード11～11の要素プロセッサ11a～11aのうち、回路接続情報を受け取ったノード（ノード11とする）の要素プロセッサ11aは、入力された回路接続情報によって示される回路を、ノード11～11の数と同数の1個の部分回路に分割する。この部分回路の分割は、例えば、特開平9-319784号公報に記載されている方法によって行う。そして、ノード11の要素プロセッサ11aは、分割した1個の部分回路に関する情報をそれぞれ、ノード11～11に相互結合網10を介し（但し、ノード11には、プロセス間通信によって）て送信する（ステップS310）。

【0074】部分回路に関する情報を受け取ったノード11～11は、それぞれが並列に動作する。これらのノード11～11の要素プロセッサ11a～11aは、それぞれ部分回路に関する情報を、後述するシミュレーション時に効率よい形式のオブジェクトデータ（部分回路を示す行列と部分回路内の各素子のパラメータ）に変換するコンパイル作業を行う（ステップS321～S321）。

【0075】更に、要素プロセッサ11a～11aは、各部分回路毎に線形連立方程式の求解のための記号処理を行う。この線形連立方程式の求解のための記号処理は、図9のフローチャートに従って行われる（ステップS331～S331）。更に、要素プロセッサ11a～11aは、それぞれステップS331～S331で記号処理した結果に対して、数値計算処理を行い、その結果を1つのノード（ここでは、ノード11とする）に相互結合網10を介して（但し、ノード11ではプロセス間通信によって）送信する（ステップS341～S341）。

【0076】各部分回路の演算結果を受信したノード11の要素プロセッサ11aは、各部分回路の縮退モデルのオブジェクトデータと結合回路情報とを用いて、結合回路のコンパイルを行い、結合回路を示す行列を生成する（ステップS350）。

【0077】次に、要素プロセッサ11aは、生成した

行列を複数の行列集合に分ける。そして、図3のフローチャートに示す処理により、複数のノードが並列動作して結合回路方程式の求解のための記号計算処理が行われる（ステップS360）。

【0078】ステップS360の処理で記号計算処理がされ、行列が圧縮格納されると、その行列を格納している局所メモリ（ここでは、局所メモリ11bとする）を有するノード11の要素プロセッサ11aは、図2のフローチャートに示すステップS140及びS150の処理を行って、数値計算処理を行う（ステップS370）。ノード11の要素プロセッサ11aは、得られた結果を波形として相互結合網10を介してワークステーション2に出力する（ステップS380）。そして、このフローチャートの処理を終了する。

【0079】以上説明したように、この実施の形態では、結合回路方程式の求解を行う場合に、複数のノード11～11での並列処理が可能となる。このため、回路シミュレーションを従来よりも高速に行うことができ、また、局所メモリ11b～11cの容量による制限も受けなくなるため、大規模な回路のシミュレーションを行うことができる。

【0080】〔実施の形態の変形〕本発明は、上記の第1、第2の実施の形態に限られず、様々な変形が可能である。以下、上記の実施の形態の変形態様について、説明する。

【0081】上記の第1の実施の形態では、ワークステーション2から処理を依頼された係数行列は予めn個の行集合に分割されているものとして説明した。しかしながら、ワークステーション2から並列コンピュータ1に処理を依頼する係数行列は、行集合に分割されていないものであってもよい。この場合、処理依頼を受け取ったノードが、並列コンピュータ1のノード11～11の数kに応じて係数行列を行集合に分割すればよい。

【0082】上記の第1、第2の実施の形態では、処理対象となるデータ（線形連立方程式の係数行列或いは回路接続情報）の投入以外のすべての処理を図1に示す並列コンピュータ1で行っていた。しかしながら、例えば、記号計算処理を並列コンピュータ1で行い、数値計算処理を並列コンピュータ1とは別のベクトル計算機で行うことも可能である。また、並列処理される部分以外をワークステーション2で実行し、並列処理される部分の処理をワークステーション20から並列コンピュータ1に依頼することも可能である。

【0083】上記の第2の実施の形態では、部分回路の線形連立方程式を求解するための記号計算処理は、部分回路毎に1つのノードで行われていた。しかしながら、この処理も、図3のフローチャートで示したのと同様に複数のノードで並列に実行するものとしてもよい。

【0084】

【発明の効果】以上説明したように、本発明によれば、

線形連立方程式の解を求める場合の記号計算において、計算時間を短くしたり、扱える行列の規模を大きくしたりすることができる。

【0085】また、大規模な回路のシミュレーションを高速に行うことができる。

【図面の簡単な説明】

【図1】本発明の第1の実施の形態に適用される並列コンピュータの構成を示すブロック図である。

【図2】本発明の第1の実施の形態にかかる線形連立方程式の求解処理の概略を示すフローチャートである。

【図3】図2の記号計算処理を詳細に示すフローチャートである。

【図4】本発明の第1の実施の形態における具体例で、ワークステーションから並列コンピュータに投入される係数行列を示す図である。

【図5】(a)～(c)は、本発明の第1の実施の形態における具体例での記号計算処理を説明する図である。

【図6】本発明の第1の実施の形態における具体例での

エントリが追加登録される順序を示す図である。

【図7】本発明の第1の実施の形態における具体例での行列の圧縮格納用データ構造を示す図である。

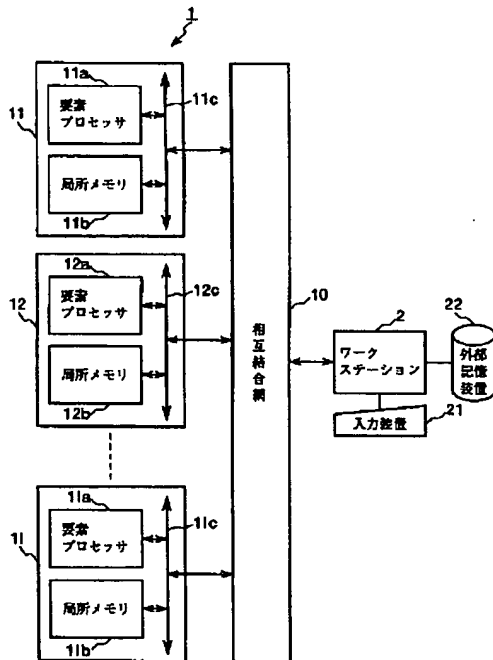
【図8】本発明の第2の実施の形態にかかる並列回路シミュレーションの処理を示すフローチャートである。

【図9】従来にかかる記号計算処理を示すフローチャートである。

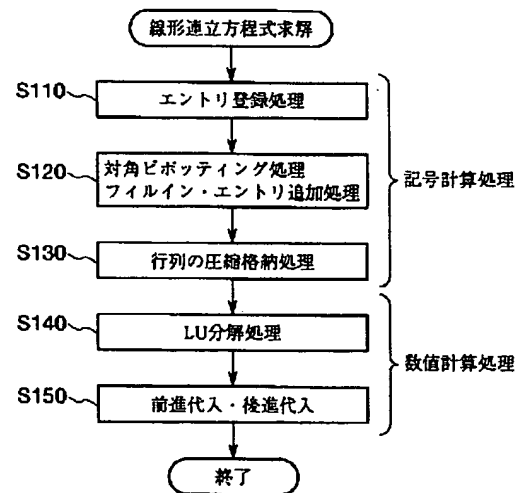
【符号の説明】

- 1 並列コンピュータ
- 10 相互結合網
- 11～111 ノード
- 11a～11a 要素プロセッサ
- 11b～11b 局所メモリ
- 11c～11c バス
- 2 ワークステーション
- 21 入力装置
- 22 外部記憶装置

【図1】



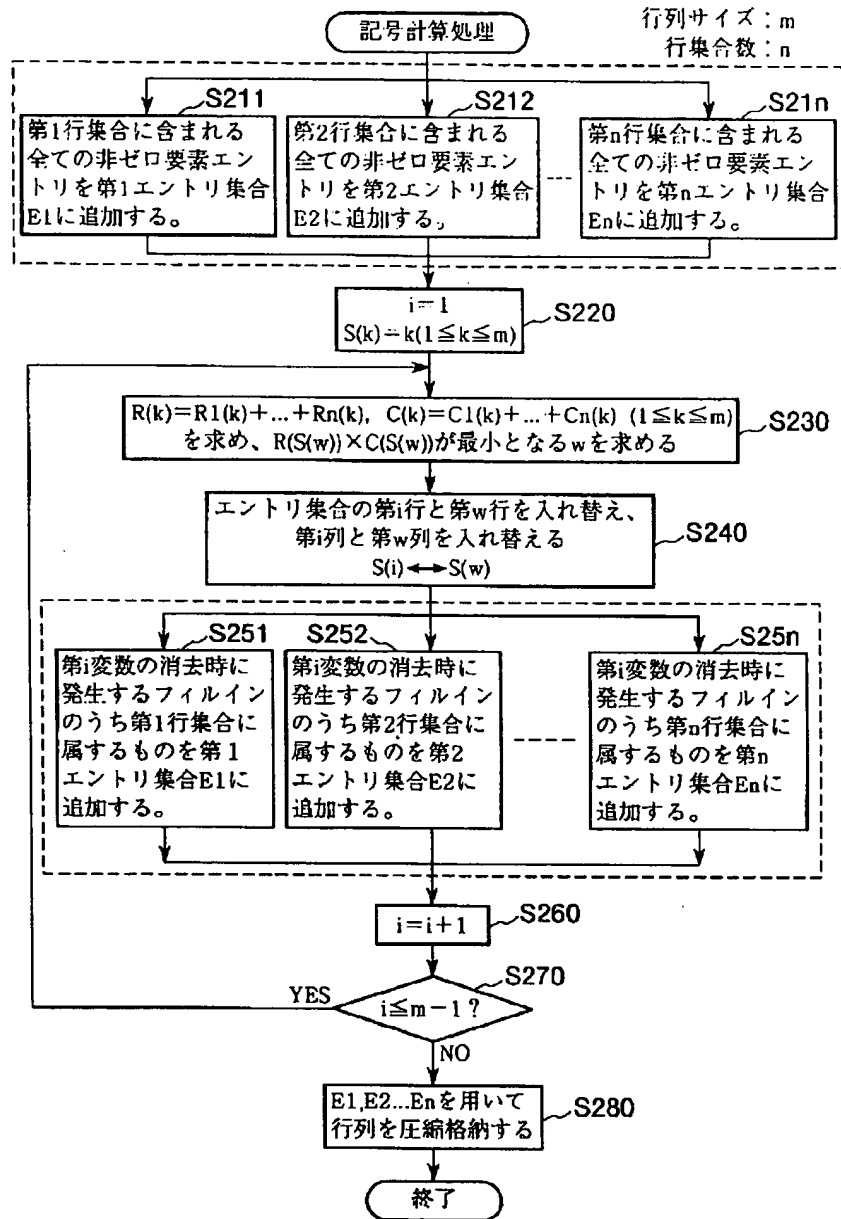
【図2】



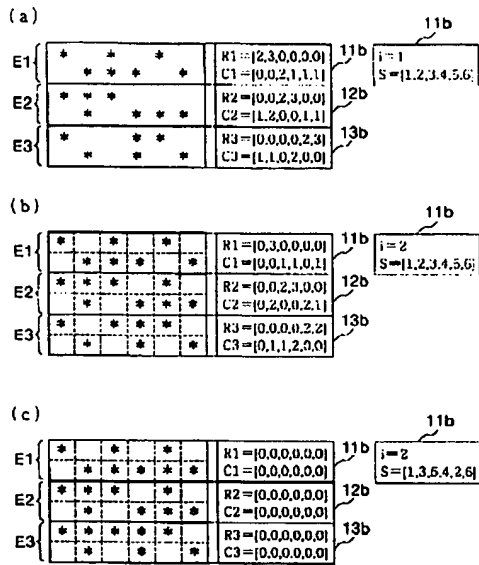
【図4】

$$\begin{Bmatrix} 2 & 0 & -1 & 0 & -1 & 0 \\ 0 & 2 & -1 & -1 & 0 & -1 \\ -1 & -1 & 3 & 0 & 0 & 0 \\ 0 & -1 & 0 & 2 & -1 & -1 \\ -1 & 0 & 0 & -1 & 2 & 0 \\ 0 & -1 & 0 & -1 & 0 & 3 \end{Bmatrix}$$

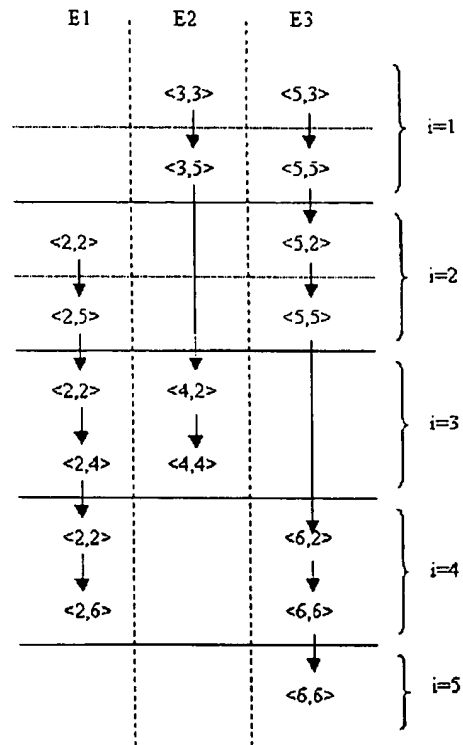
【図3】



【図 5】



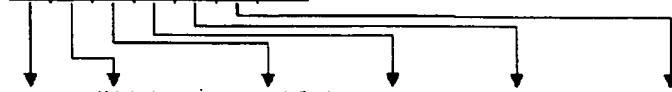
【図 6】



【図 7】

ROWPTR (各行の先頭エントリのインデックス)

1	4	9	13	17	22	25
---	---	---	----	----	----	----



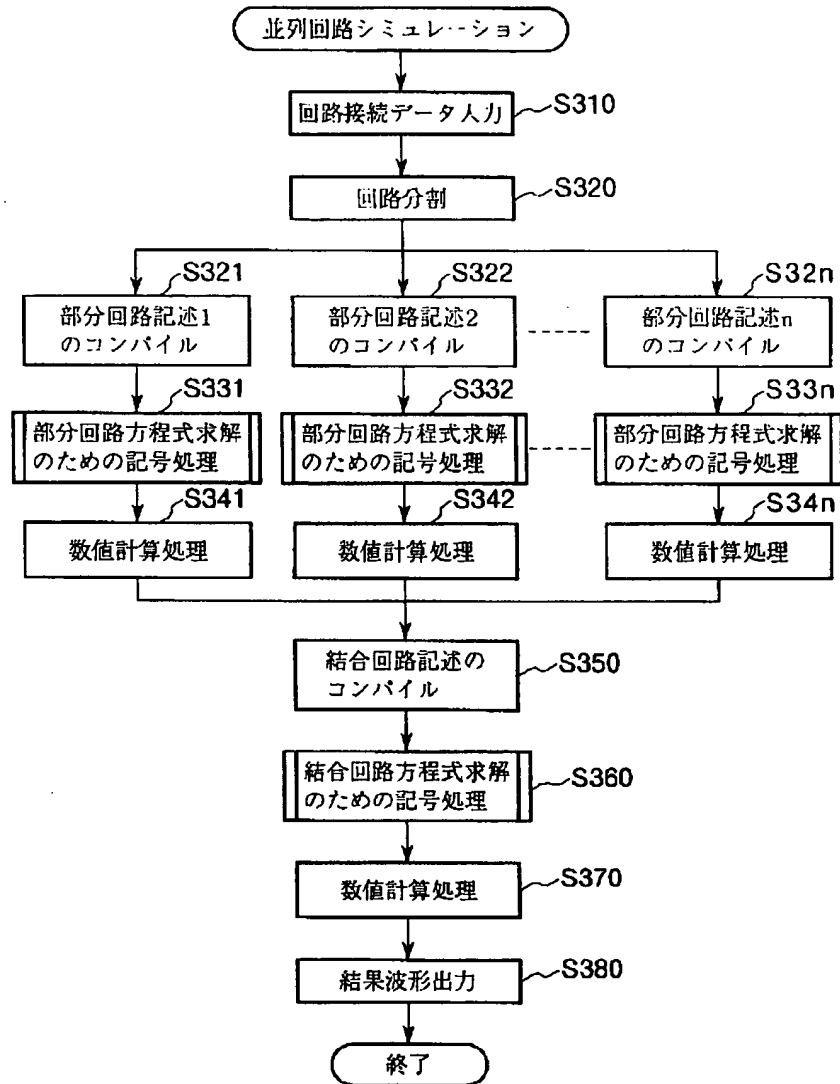
COLIDX (対応するエントリの列番号)

1	3	5	2	3	4	5	6	1	2	3	5	2	4	5	6	1	2	3	4	5	2	4	6
---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

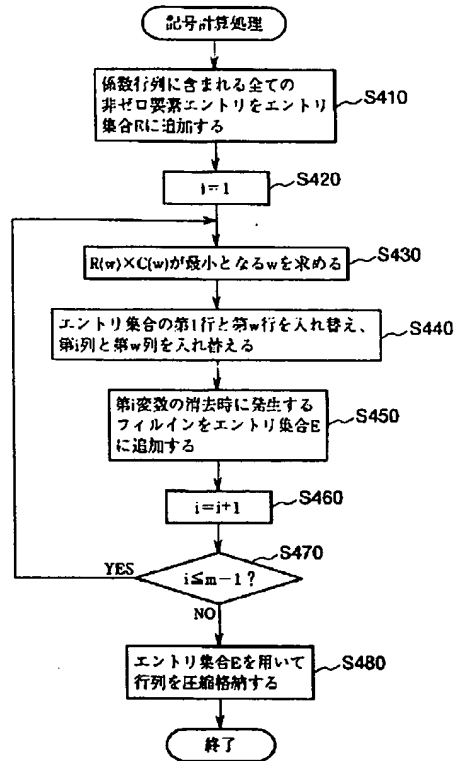
2	-1	-1	2	-1	-1	0	-1	-1	-1	3	0	-1	2	-1	-1	0	0	-1	2	-1	-1	3
---	----	----	---	----	----	---	----	----	----	---	---	----	---	----	----	---	---	----	---	----	----	---

VALUE (対応するエントリの行列要素の値)

【図8】



【図 9】



**This Page is Inserted by IFW Indexing and Scanning
Operations and is not part of the Official Record**

BEST AVAILABLE IMAGES

Defective images within this document are accurate representations of the original documents submitted by the applicant.

Defects in the images include but are not limited to the items checked:

- ☐ BLACK BORDERS
- ☐ IMAGE CUT OFF AT TOP, BOTTOM OR SIDES
- ☐ FADED TEXT OR DRAWING
- ☐ BLURRED OR ILLEGIBLE TEXT OR DRAWING
- ☐ SKEWED/SLANTED IMAGES
- ☐ COLOR OR BLACK AND WHITE PHOTOGRAPHS
- ☐ GRAY SCALE DOCUMENTS
- ☐ LINES OR MARKS ON ORIGINAL DOCUMENT
- ☐ REFERENCE(S) OR EXHIBIT(S) SUBMITTED ARE POOR QUALITY
- ☐ OTHER: _____

IMAGES ARE BEST AVAILABLE COPY.

As rescanning these documents will not correct the image problems checked, please do not report these problems to the IFW Image Problem Mailbox.